THE UNIVERSITY OF SYDNEY

COMP5416 Lab 2

HTTP Wireshark

The goal of this tutorial is to analyse the HTTP protocol requests. To this end, you will get familiar with Wireshark, an open source network protocol analyser, that resembles **`tcpdump`** with a graphical user interface.

Make sure your machine is running Linux or reboot it.

Before starting make sure you can access the web using a browser through the proxy. If not, you can either change Firefox's network preferences or setup the environment variable before starting firefox **`export http_proxy=http://www-cache.it.usyd.edu.au:8000; firefox &`**.

Exercise 1: Basic GET/response interaction

Let's begin our exploration of HTTP by downloading a very simple HTML file - one that is very short, and contains no embedded objects. Do the following:

1. Start up the Wireshark packet sniffer by clicking on **`Applications > Internet > Wireshard Network Analyzer`** but do not yet begin packet capture. Enter **`http`** in the filter specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.

2. Start up a web browser, preferably Firefox.

3. Wait about one minute before beginning Wireshark packet capture, by clicking the small icon on the top-left of the Wireshard window. (Select to capture from the interface that corresponds to the external IP address.)

4. Enter the following to your browser http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html. Your browser should display the very simple, one-line HTML file.

5. Stop Wireshark packet capture by clicking the icon with a white cross on a red disk in the Wireshark bar.

Your Wireshark window should look similar to the window shown in Figure 1. If you are unable to run Wireshark on a live network connection, you can download a packet trace that was created when the steps above were followed.[1]

[1]Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file http-ethereal-trace-1. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can

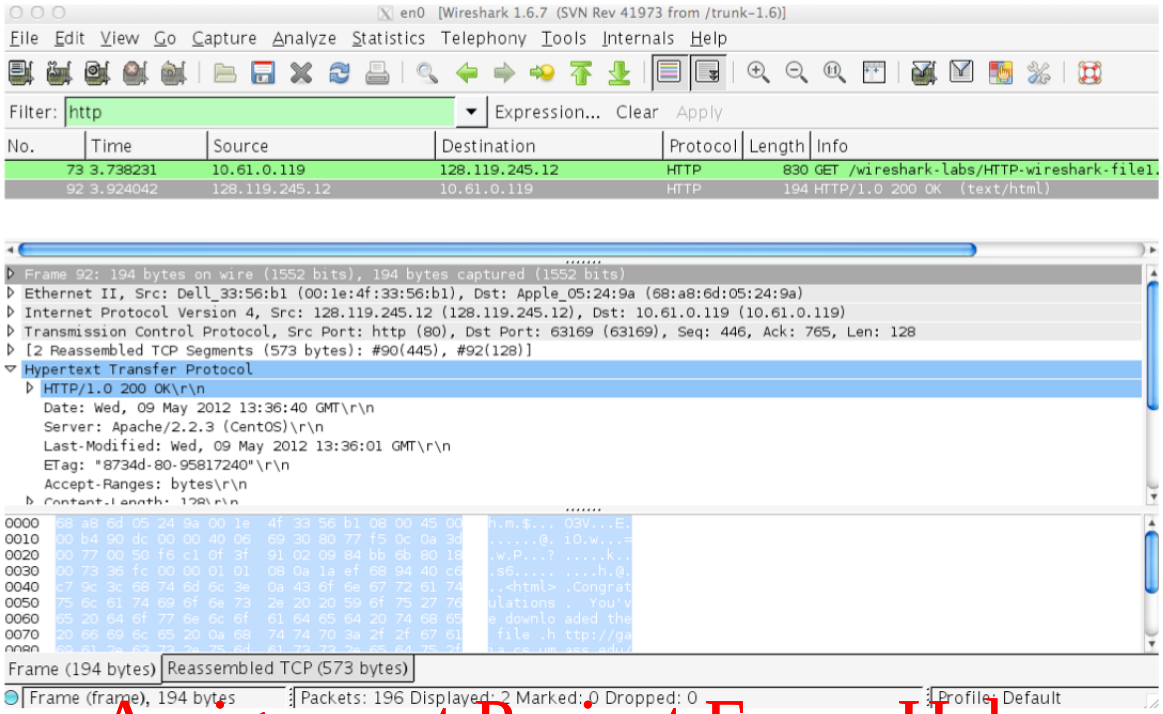

Figure 1: Wireshark display after http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html has been retrieved by your browser

The example in Figure 1 shows in the packet-listing window that two HTTP messages were captured: the GET message (from your browser to the gaia.cs.umass.edu web server) and the response message from the server to your browser. The packet-contents window shows details of the selected message (in this case the HTTP OK message, which is highlighted in the packet-listing window). Recall that since the HTTP message was carried inside a TCP segment, which was carried inside an IP datagram, which was carried within an Ethernet frame, Wireshark displays the Frame, Ethernet, IP, and TCP packet information as well. We want to minimize the amount of non-HTTP data displayed (we are interested in HTTP here), so make sure the boxes at the far left of the Frame, Ethernet, IP and TCP information have a right-pointing triangle (which means there is hidden, undisplayed information), and the HTTP line has a down-pointing triangle (which means that all information about the HTTP message is displayed).

(Note: You should ignore any HTTP GET and response for favicon.ico. If you see a reference to this file, it is your browser automatically asking the server if the server has a small icon file that should be displayed next to the displayed URL in your browser. We'll ignore references to this pesky file in this lab.).

By looking at the information in the HTTP GET and response messages, answer the following questions.

1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?

load it into Wireshark and view the trace using the File pull down menu, choosing Open, and then selecting the http-ethereal-trace-1 trace file. The resulting display should look similar to Figure 1. (The Wireshark user interface displays just a bit differently on different operating systems, and in different versions of Wireshark).

What languages (if any) does your browser indicate that it can accept from the server?

2. What is the IP address of your computer? Of the gaia.cs.umass.edu server?
3. What is the status code returned from the server to your browser?
4. When was the HTML file that you are retrieving last modified at the server?
5. How many bytes of content are being returned to your browser?
6. By inspecting the raw data in the packet content window, do you see any headers within the data that are not displayed in the packet-listing window? If so, name one.

In your answer to question 5 above, you might have been surprised to find that the document you just retrieved was last modified within a minute before you downloaded the document. That's because (for this particular file), the **`gaia.cs.umass.edu`** server is setting the file's last-modified time to be the current time, and is doing so once per minute. Thus, if you wait a minute between accesses, the file will appear to have been recently modified, and hence your browser will download a "new" copy of the document.

Exercise 2: Conditional GET/response interaction

Recall (cf. textbook, Section 2.2.6) that most web browsers perform object caching and thus perform a conditional GET when retrieving an HTTP object. Before performing the steps below, make sure your browser's cache is empty. To do this under Firefox, select **`Edit > Preferences > Clear Recent History`** and check the cache box (for Internet Explorer, select Tools>Internet Options>Delete File). These actions will remove cached files from your browser's cache. Now do the following:

- Restart your web browser, and make sure your browser's cache is cleared, as discussed above.
- Start up the capture of Wireshark packet sniffer.
- Enter the following URL into your browser. http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html. Your browser should display a very simple five-line HTML file.
- Quickly (within few seconds) click the refresh button on your browser or press F5.
- Stop Wireshark packet capture, and enter **`http`** in the filter window, so that only captured HTTP messages will be displayed later in the packet-listing window.
- (Note: If you are unable to run Wireshark on a live network connection, you can use the http-ethereal-trace-2 packet trace to answer the questions below; see footnote 1. This trace file was gathered while performing the steps above on one of the author's computers.)

Answer the following questions:

1. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?

2. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?

3. Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE:" line in the HTTP GET? If so, what information follows the "IF-MODIFIED-SINCE:" header?

4. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

Exercise 3: Authentication

Finally, let's try visiting a web site that is password-protected and examine the sequence of HTTP message exchanged for such a site. The URL http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html is password protected. The username is **`wireshark-students`** , and the password is **`network`**. So let's access this "secure" password-protected site. Do the following:

- Make sure your browser's cache is cleared, as discussed above, and restart your browser.
- Restart capturing with Wireshark packet sniffer.
- Enter the following URL into your browser http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html.

Type the requested user name and password into the pop up box.

- Stop Wireshark packet capture, and enter **`http`** in the filter window, so that only captured HTTP messages will be displayed later in the packet-listing window.
- (Note: If you are unable to run Wireshark on a live network connection, you can use the http-ethereal-trace-5 packet trace to answer the questions below; see footnote 2. This trace file was gathered while performing the steps above on one of the author's computers.)

Now let's examine the Wireshark output. You might want to first read up on HTTP authentication by reviewing the easy-to-read material on "HTTP Access Authentication Framework" at http://frontier.userland.com/stories/storyReader$2159. Answer the following questions:

1. What is the server's response (status code and phrase) in response to the initial HTTP GET message from your browser?

2. When your browser's sends the HTTP GET message for the aforementioned webpage for the second time, what new field is included in the HTTP GET message? The username (wireshark-students) and password (network) that you entered are encoded in the string of characters (d2lyZXNoYXJrLXN0dWRlbnRzOm5ldHdvcms=) following the "Authorization: Basic" header in the client's HTTP GET message. While it may appear that your username and password are encrypted, they are simply encoded in a format known as Base64 format. The username and password are not encrypted. To see this, go to http://www.motobit.com/util/base64-decoder-encoder.asp and enter the base64-encoded string d2lyZXNoYXJrLXN0dWRlbnRz

and decode. Voila! You have translated from Base64 encoding to ASCII encoding, and thus should see your username! To view the password, enter the remainder of the string Om5ldHdvcms= and press decode.